UNIVERSIDAD SAN FRANCISCO DE QUITO USFQ

Colegio de Ciencias e Ingenierías

DESARROLLO DE UN BOT TUTOR DE AJEDREZ MEJORADO MEDIANTE LA INTEGRACIÓN DE UN LLM PARA ASISTENCIA Y ANÁLISIS

Andrés Sol Martínez Sotomayor

Ingeniería en las Ciencias de la Computación

Trabajo de fin de carrera presentado como requisito previo a la obtención del título de ingeniero en Ciencias de la Computación

Quito, 15 de diciembre de 2024

UNIVERSIDAD SAN FRANCISCO DE QUITO USFQ

Colegio de Ciencias e Ingenierías

HOJA DE CALIFICACIÓN
DE TRABAJO DE FIN DE CARRERA

Desarrollo de un Bot tutor de ajedrez mejorado mediante la integración de un LLM para asistencia y análisis

Andrés Sol Martínez Sotomayor

Nombre del profesor, Título académico **Roberto Andrade, Ph. D**

Quito, 15 de diciembre de 2024

© DERECHOS DE AUTOR

Nombres y apellidos: Andrés Sol Martínez Sotomayor

Código: 00213046

Cédula de identidad: 1722041215

Lugar y fecha: Quito, 14 de diciembre de 2024

ACLARACIÓN PARA PUBLICACIÓN

Nota: El presente trabajo, en su totalidad o cualquiera de sus partes, no debe ser considerado como una publicación, incluso a pesar de estar disponible sin restricciones a través de un repositorio institucional. Esta declaración se alinea con las prácticas y recomendaciones presentadas por el Committee on Publication Ethics COPE descritas por Barbour et al. (2017) Discussion document on best practice for issues around theses publishing, disponible en http://bit.ly/COPETheses.

UNPUBLISHED DOCUMENT

Note: The following capstone project is available through Universidad San Francisco de Quito USFQ institutional repository. Nonetheless, this project – in whole or in part – should not be considered a publication. This statement follows the recommendations presented by the Committee on Publication Ethics COPE described by Barbour et al. (2017) Discussion document on best practice for issues around theses publishing available on http://bit.ly/COPETheses.

RESUMEN

Este trabajo presenta el desarrollo de un innovador bot tutor de ajedrez que integra el motor de ajedrez Maia con el modelo de lenguaje Claude 3.5 Haiku para crear una experiencia de enseñanza personalizada y accesible. La motivación surge de la brecha existente entre la capacidad de juego de los motores de ajedrez tradicionales y su habilidad para explicar decisiones de manera comprensible, especialmente en contextos como el ecuatoriano donde el acceso a instrucción de alto nivel es limitado.

El sistema implementa una arquitectura modular que combina un backend en Django con una interfaz web interactiva, incorporando técnicas avanzadas de procesamiento de lenguaje natural para la interpretación de movimientos y análisis posicional. La integración del motor Maia, especializado en emular el juego humano, con el LLM permite generar explicaciones contextualizadas y pedagógicamente efectivas de conceptos ajedrecísticos complejos. Durante el desarrollo, la transición del modelo LLaMa 3.2 al Claude 3.5 Haiku mejoró significativamente el rendimiento del sistema, reduciendo los tiempos de respuesta de 17 minutos a segundos. Los resultados demuestran el potencial de los sistemas de tutoría inteligente en el ajedrez, aunque también revelan limitaciones en la capacidad actual de los LLMs para proporcionar instrucción verdaderamente personalizada. El proyecto establece bases para futuros desarrollos en sistemas educativos basados en IA, sugiriendo la necesidad de incorporar seguimiento del progreso del usuario y metodologías especializadas para diferentes aspectos del aprendizaje del ajedrez.

Palabras clave: Inteligencia Artificial, Ajedrez, Sistemas de Tutoría Inteligente, Procesamiento de Lenguaje Natural, Aprendizaje Automatizado, Modelos de Lenguaje Grandes, Educación Personalizada, Python, Django.

ABSTRACT

This work presents the development of an innovative chess tutor bot that integrates the Maia chess engine with the Claude 3.5 Haiku language model to create a personalized and accessible teaching experience. The motivation arises from the gap between traditional chess engines' playing capabilities and their ability to explain decisions comprehensibly, especially in contexts like Ecuador where access to high-level instruction is limited.

The system implements a modular architecture that combines a Django backend with an interactive web interface, incorporating advanced natural language processing techniques for move interpretation and positional analysis. The integration of the Maia engine, specialized in emulating human play, with the LLM enables the generation of contextualized and pedagogically effective explanations of complex chess concepts. During development, the transition from LLaMa 3.2 model to Claude 3.5 Haiku significantly improved system performance, reducing response times from 17 minutes to seconds. The results demonstrate the potential of intelligent tutoring systems in chess, while also revealing limitations in the current capacity of LLMs to provide truly personalized instruction. The project establishes foundations for future developments in AI-based educational systems, suggesting the need to incorporate user progress tracking and specialized methodologies for different aspects of chess learning.

Key words: Artificial Intelligence, Chess, Intelligent Tutoring Systems, Natural Language Processing, Machine Learning, Large Language Models, Personalized Education, Python, Django.

TABLA DE CONTENIDO

ÍNDICE DE FIGURAS

INTRODUCCIÓN

En la intersección de la inteligencia artificial y los juegos de estrategia, este trabajo de tesis se enfoca en el desarrollo de un innovador bot tutor de ajedrez que integra un Modelo de Lenguaje Grande (LLM) para asistencia y análisis. El ajedrez, desde hace tiempo un campo de prueba ideal para los avances en IA ha sido testigo de hitos significativos, desde la victoria de Deep Blue sobre Kaspárov hasta los logros de Alpha Zero. Sin embargo, persiste una brecha entre la capacidad de juego de estos sistemas y su habilidad para enseñar efectivamente. Esta evolución tecnológica, aunque impresionante en términos de capacidad de juego, ha creado una paradoja en la enseñanza del ajedrez: mientras más fuertes se vuelven los motores, más difícil resulta para los estudiantes comprender su razonamiento.

Los motores de ajedrez tradicionales, aunque extremadamente fuertes, a menudo carecen de la capacidad de explicar sus decisiones de manera comprensible para los jugadores humanos, especialmente para aquellos en proceso de aprendizaje. Esta limitación se hace más evidente en contextos como el ecuatoriano, donde el ajedrez, aunque respetado, no ha alcanzado los niveles de desarrollo observados en otros países. A 2024, Ecuador cuenta con solo 3 Grandes Maestros y 17 Maestros Internacionales reconocidos por la FIDE, lo que subraya la necesidad de herramientas de entrenamiento accesibles y efectivas.

Nuestro proyecto busca abordar esta brecha combinando la fuerza de juego de un motor de ajedrez moderno con la capacidad de comunicación y explicación de un LLM. El objetivo es crear un "Gran Maestro virtual" accesible para cualquier jugador, independientemente de su ubicación o recursos. Este bot tutor no solo jugará al ajedrez, sino que también proporcionará análisis detallados, responderá preguntas y ofrecerá insights estratégicos de manera comprensible. En las siguientes secciones, exploraremos el diseño e implementación de este bot tutor de ajedrez. Comenzaremos con una revisión del estado del arte en motores de ajedrez

y LLMs, seguida de una descripción de la arquitectura del sistema. Abordaremos los componentes clave del bot, incluyendo la integración del motor de ajedrez, la implementación del LLM para análisis y comunicación, y el desarrollo de la interfaz de usuario. También discutiremos los desafíos técnicos encontrados, como la interpretación precisa de las evaluaciones del motor por parte del LLM y la generación de explicaciones coherentes.

Finalmente, presentaremos los resultados de nuestras pruebas, discutiremos las implicaciones de nuestro trabajo y exploraremos direcciones futuras para la investigación en este campo. Este proyecto no solo representa un avance en el entrenamiento de ajedrez asistido por IA, sino que también establece un precedente para futuros sistemas de tutoría inteligente en diversos dominios del conocimiento.

Desarrollo del Tema

ESTADO DEL ARTE

Históricamente los motores de ajedrez fueron creados usando un gran número de feature selection para guiar la selección de movimientos, usando grandes cantidades de conocimiento de dominio especifico, sin embargo más recientemente motores de ajedrez alternativos como Giraffe [Lai 2015] y proyectos como Alpha Zero el motor más fuerte que existe actualmente ofrecieron los primeros modelos y redes neuronales especializados en obtener un nivel de juego igual al de un gran maestro a través de métodos supervisados y no supervisados de entrenamiento se buscó en su lugar un motor que simule la forma en la que juega un jugador humano. Tal modelo es Maia, un motor basado en un framework de deep learning entrenado en juegos humanos para buscar producir el movimiento que un jugador humano jugaría con un éxito sobre el 50% (mucho más arriba que la mayoría de los motores).

Es en los últimos años el desarrollo de inteligencias artificiales generales ha llevado a que existan agentes generales de juegos como MuZero [Schrittwieser et al, 2019] de los mismo creadores de Alpha Zero que es un modelo que no conoce las reglas y puede jugar una variedad de juegos, o modelos como SentiMate [Kamlish, 2019] que usan LSTM bidireccionales para generar análisis de sentimiento para entrenar un CNN que es capaz de jugar mejor que un jugador aleatorio o un motor de DeepChess con profundidad 1. Sin embargo, es el crecimiento de los LLMs que lleva a una nueva generación de motores de ajedrez que usan diferentes capacidades para aprender y jugar ajedrez, como ChessLLM que usa un instruct de GPT-3.5-trubo-instruct para generar probabilísticamente un movimiento [Nicolas, 2023]. Sin embargo, cabe recalcar que los LLM, aunque pueden ser instruidos a jugar movimientos, con la condición

de proporcionarle los movimientos legales primero para evitar alucinaciones, un LLM no juega con la intención de ganar. Por ejemplo, si se pone como oponente a un jugador aleatorio, es decir un bot que juega un movimiento al azar (al cual cualquier jugador humano podría ganarle) nunca es capaz de ganar (Saplin, 2024). Similarmente varios de los modelos grandes y comercialmente disponibles pueden ser modificados o reentrenados para generar movimientos óptimos, pero pierden la capacidad de comunicarse con el usuario y no son capaces de ver más adelante del primero movimiento. La eficacia de estos motores de ajedrez depende mucho en si se juega se trata de juegos similares a juegos populares o conocidos, el formato en el que se ingresen los movimientos y cuánto tiempo se le dé al modelo para pensar.

Player	Draws ▼	Wins	Mistakes	Tokens
gpt-4-turbo-2024-04-09	93.33%	0.00%	0.00	6.03
gpt-4o-2024-08-06	90.00%	0.00%	0.00	6.58
gpt-4o-2024-05-13	83.33%	0.00%	0.72	28.41
anthropic.claude-v3-5-sonnet	73.33%	0.00%	2.33	81.07
gpt-4o-mini-2024-07-18	60.00%	0.00%	8.26	108.22
llama-3-70b-instruct-awq	50.00%	0.00%	9.67	41.61
gemini-1.5-pro-preview-0409	36.67%	0.00%	29.32	13.41
gemini-1.5-flash-001	33.33%	0.00%	21.39	19.91
gemma-2-27b-it@q6_k_l	26.67%	0.00%	18.36	55.04
gemma-2-9b-it-8bit	13.33%	0.00%	36.63	58.12
anthropic.claude-v3-haiku	0.00%	0.00%	9.80	205.71
gpt-35-turbo-0125	0.00%	0.00%	1046.51	82.02
llama-3.1-8b-instruct-fp16	0.00%	0.00%	96.38	71.86
Random Player (as White)	89.50%	10.50%	0.00	NaN
Random Player (as Black)	89.50%	0.00%	0.00	NaN
Chess engine (as Black)	0.00%	100.00%	0.00	NaN

Figura 1: Comparación del rendimiento de varios modelos de lenguaje y jugadores aleatorios en ajedrez.

PROPUESTA

El presente proyecto propone el desarrollo de un Bot tutor de ajedrez mejorado mediante la integración de un Modelo de Lenguaje Grande (LLM) para asistencia y análisis. Esta innovación representa un avance significativo en el campo del entrenamiento de ajedrez asistido por IA, abordando una necesidad crucial en el aprendizaje del juego: la capacidad de comprender y contextualizar las sugerencias de los motores de ajedrez de manera accesible para jugadores humanos. La relevancia de este proyecto se amplifica en el contexto ecuatoriano, donde la escasez de Grandes Maestros y Maestros Internacionales (3 y 17 respectivamente en 2024) crea una barrera significativa para el desarrollo del ajedrez de alto nivel. La integración de un LLM con capacidades explicativas avanzadas junto a un motor de ajedrez especializado no solo democratiza el acceso a instrucción de calidad, sino que también establece un nuevo paradigma en la enseñanza personalizada del ajedrez, adaptándose a las necesidades específicas de cada estudiante y al contexto cultural local.

Este proyecto se guía en los principios de los sistemas de tutoría inteligente, un campo pionero en el que Anderson et al. demostraron el potencial de la tecnología para proporcionar instrucción individualizada y adaptativa. Como señalan estos autores, los tutores inteligentes buscan "involucrar al estudiante en una actividad de razonamiento sostenida e interactuar con el estudiante basándose en una comprensión profunda del comportamiento del estudiante" (Anderson et al., 1985). Siguiendo este enfoque, nuestro Bot de ajedrez integra la precisión analítica de un motor de ajedrez con la capacidad explicativa de un LLM, creando así un entorno de aprendizaje que se adapta dinámicamente a las necesidades y el nivel de cada jugador. Esta combinación permite emular la experiencia de tener un tutor humano experto, proporcionando retroalimentación inmediata y ajustando la instrucción a las necesidades individuales del estudiante.

La integración de un LLM con un motor de ajedrez tradicional permitirá al Bot no solo sugerir movimientos, sino también proporcionar explicaciones detalladas, responder a preguntas específicas y ofrecer insights estratégicos, emulando la experiencia de tener un entrenador personal de ajedrez. El Bot propuesto incluirá un sistema de ELO adaptativo que ajustará dinámicamente su nivel de juego al del usuario, ofreciendo un entorno de entrenamiento óptimo para jugadores de todos los niveles. Además, incorporará un módulo de análisis automatizado capaz de revisar partidas en tiempo real o importadas, proporcionando explicaciones detalladas sobre movimientos clave, errores y oportunidades perdidas. Una unidad lógica avanzada potenciará las capacidades del LLM para comprender y contextualizar el análisis del motor de ajedrez, utilizando técnicas como parsing avanzado, análisis heurístico de la posición y reconocimiento de patrones. Esta innovación permitirá al Bot ofrecer insights profundos y personalizados, elevando la experiencia de aprendizaje a un nivel sin precedentes. Este proyecto no solo busca avanzar en el campo del ajedrez y la IA, sino que también establece un precedente para futuros sistemas de tutoría inteligente en diversos dominios del conocimiento. Al crear un tutor de ajedrez IA capaz de explicar estrategias complejas de manera comprensible, esencialmente desarrollamos un "Gran Maestro virtual" accesible para cualquier jugador, con el potencial de revolucionar la forma en que se aprende y enseña el ajedrez en el siglo XXI.

ARQUITECTURA DEL SISTEMA

La implementación del Bot Tutor de Ajedrez se fundamenta en una arquitectura modular y escalable que permite la integración eficiente de múltiples componentes especializados, cada uno diseñado para abordar aspectos específicos del proceso de tutoría. La arquitectura del sistema del Bot Tutor de Ajedrez se basa en un diseño modular que integra componentes de backend y frontend para crear una experiencia de usuario interactiva y

eficiente. En el núcleo del sistema se encuentra el Sistema de Lógica de Ajedrez, responsable de gestionar el estado del juego, validar movimientos y comunicarse con el motor de ajedrez Maia. Este sistema utiliza la librería python-chess para la representación del tablero y la generación de movimientos legales, mientras que la integración con Maia se realiza a través de una capa de abstracción que facilita la comunicación entre los componentes.

El Sistema de Lógica de Ajedrez se complementa con un módulo de Gestión de Modelos, que se encarga de la interacción con el modelo de lenguaje (LLM) utilizado para el análisis y la generación de respuestas. Este módulo emplea técnicas avanzadas de procesamiento de lenguaje natural, como la clasificación de intenciones y la extracción de movimientos, para interpretar las consultas del usuario y generar respuestas coherentes y relevantes. La integración del LLM se realiza mediante una arquitectura de prompts dinámicos, que permite adaptar las respuestas al contexto específico de cada interacción.

En el frontend, la interfaz de usuario se diseñó con el objetivo de crear una experiencia intuitiva y atractiva para los usuarios. El componente central es un tablero de ajedrez interactivo, que permite a los usuarios realizar movimientos y visualizar el estado actual del juego. Este tablero se implementa utilizando tecnologías web modernas, como HTML5 y JavaScript, y se comunica con el backend a través del framework de Django. Además del tablero, la interfaz de usuario incluye un panel lateral que muestra el historial de movimientos y un chat interactivo donde los usuarios pueden hacer preguntas y recibir respuestas del Bot Tutor. El diseño de la interfaz se centra en la usabilidad y la accesibilidad, con una disposición clara de los elementos y un esquema de colores.

Figura 2: Primera versión de la interfaz de usuario, a la izquierda un tablero interactivo, a la derecha arriba el listado de movimientos, abajo el chat con el tutor de ajedrez.

El flujo de interacción entre el usuario y el sistema se basa en un modelo de comunicación bidireccional. Los usuarios pueden realizar movimientos en el tablero, que se envían al backend para su validación y procesamiento. A su vez, el backend genera respuestas y sugerencias de movimientos, que se presentan al usuario a través de la interfaz de chat. Este flujo de interacción se optimiza mediante técnicas de comunicación en tiempo real, como WebSockets. La gestión del estado del juego es un aspecto crítico de la arquitectura del sistema. El backend mantiene una representación centralizada del estado del tablero, que se actualiza con cada movimiento realizado por el usuario o generado por el motor Maia. Este estado se sincroniza con el frontend a través de actualizaciones periódicas, lo que garantiza que la interfaz de usuario siempre muestre una representación precisa del juego en curso.

Además de la sincronización del estado del juego, la arquitectura del sistema también debe gestionar la comunicación entre los diferentes componentes. Esto se logra mediante una

combinación de protocolos y formatos de datos estándar, como JSON para el intercambio de datos estructurados y FEN (Notación Forsyth-Edward) para la representación del estado del tablero. En resumen, la arquitectura del Bot Tutor de Ajedrez se basa en un diseño modular y escalable que integra componentes de backend y frontend para crear una experiencia de usuario rica e interactiva. Desde el Sistema de Lógica de Ajedrez y la integración con el motor Maia en el backend, hasta el diseño intuitivo de la interfaz de usuario y el flujo de interacción optimizado en el frontend, cada componente se desarrolla con un enfoque en la eficiencia, la usabilidad y la capacidad de respuesta. Esta arquitectura sienta las bases para un sistema robusto y flexible, capaz de adaptarse a las necesidades de los usuarios y ofrecer una experiencia de aprendizaje de ajedrez única y efectiva.

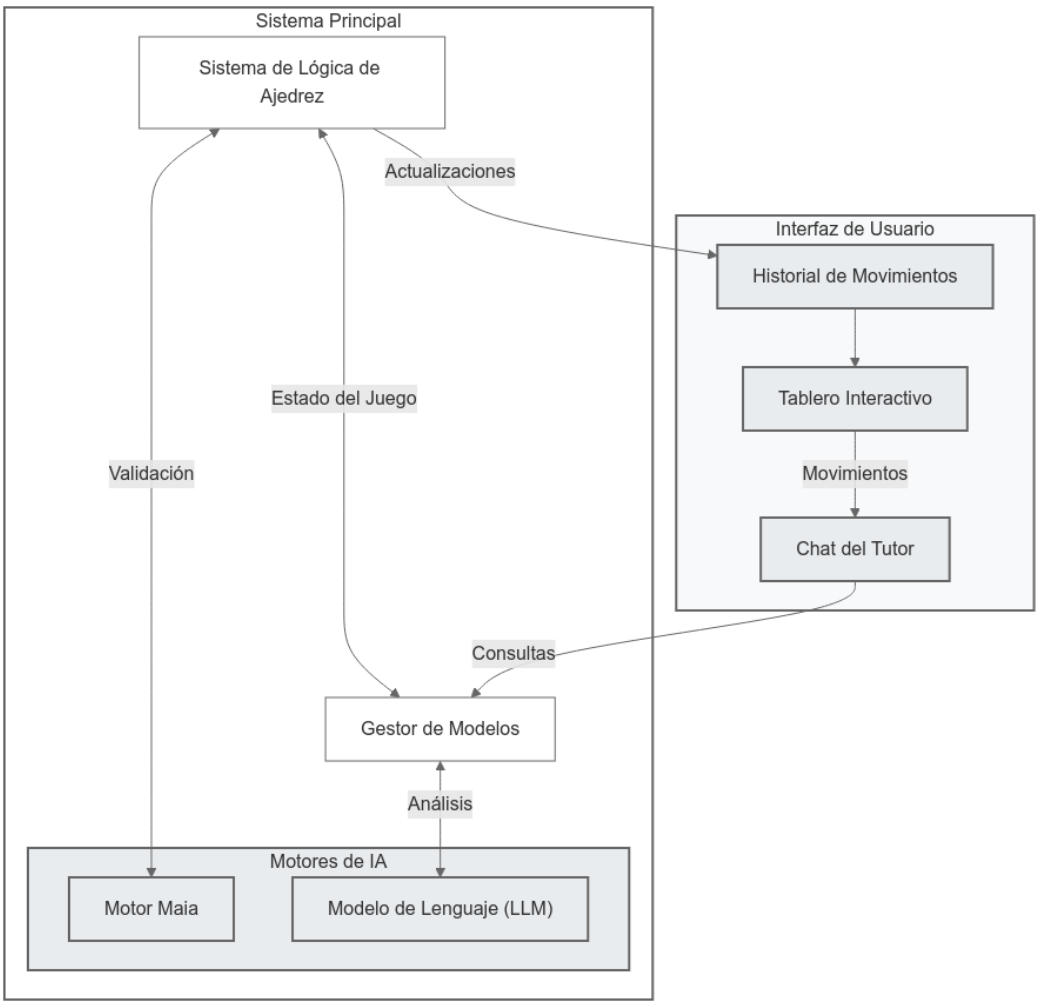

Figura 3: Diagrama de la arquitectura del sistema Bot Tutor de Ajedrez.

MEJORANDO EL ANÁLISIS SOBRE AJEDREZ QUE UN LLM PUEDE PRODUCIR

Con el fin de dar el mayor contexto posible a LLM al momento de explicar el razonamiento del motor detrás de sus movimientos se extraerá conocimiento de los movimientos mediante métodos heurísticos y de análisis sobre el motor ya existente.

El parsing avanzado de movimientos es fundamental para transformar la salida cruda del motor de ajedrez en información estructurada y comprensible para el LLM. Este proceso se implementará mediante un algoritmo de análisis sintáctico que no solo interpretará la notación algebraica estándar, sino que también desentrañará las sutilezas y complejidades de cada movimiento. El sistema comenzará por tokenizar la entrada del motor, identificando cada componente individual de los movimientos (pieza, casilla de origen, casilla de destino, capturas, etc.). Luego, utilizará una gramática formal especialmente diseñada para el ajedrez, que incluirá reglas para reconocer patrones complejos como el enroque (tanto corto como largo), capturas al paso, y promociones de peones. El parsing también incorporará un análisis de la calidad de los movimientos basado en los cambios en la evaluación del motor. Se desarrollará un algoritmo que compare las evaluaciones antes y después de cada movimiento, categorizándolos como excelentes, buenos, imprecisos o errores, proporcionando al LLM una comprensión matizada de la calidad de cada decisión en el contexto de la partida. Este mismo parsing en reverso se puede usar para transformar los movimientos que del usuario al LLM en lenguaje natural a algo que el motor de ajedrez pueda entender también.

La descomposición del análisis del motor es un proceso crucial que transformará la densa salida del motor en componentes discretos y manejables para el LLM. Este proceso se implementará mediante un algoritmo de segmentación jerárquica que operará en múltiples niveles de granularidad. En el nivel más alto, el algoritmo identificará y separará las líneas de variación principales de las secundarias. Esto se logrará mediante el análisis de la estructura de árbol de las variantes, donde la línea principal será aquella con la evaluación más favorable para el jugador al turno, y las líneas secundarias serán ramificaciones alternativas. Dentro de cada variante, el algoritmo identificará puntos críticos. Estos se definirán como posiciones donde ocurren cambios significativos en la evaluación, o donde se toman decisiones estratégicas importantes. Para determinar estos puntos, se implementará un sistema de umbral adaptativo que comparará las evaluaciones de movimientos consecutivos y detectará saltos significativos. Las evaluaciones numéricas proporcionadas por el motor se aislarán y se contextualizarán. Esto implicará no solo extraer el valor numérico, sino también interpretarlo en relación con la fase de la partida (apertura, medio juego, final) y el estilo de juego (posicional vs táctico). Se desarrollará un módulo de interpretación que traducirá estos valores numéricos a descripciones cualitativas comprensibles para el LLM.

La capa de análisis heurístico se implementará como un sistema experto basado en reglas, enriquecido con técnicas de aprendizaje automático. Esta capa operará en paralelo con la evaluación numérica del motor, proporcionando una perspectiva cualitativa y estratégica que complementa el cálculo puro. El análisis del balance material irá más allá de la simple suma de los valores de las piezas. Se implementará un sistema de ponderación dinámica que ajustará el valor de cada pieza basándose en factores posicionales. Por ejemplo, un caballo en una posición central y estable podría valorarse más que un alfil atrapado detrás de sus propios

peones. Este sistema utilizará una red neuronal entrenada en miles de partidas de alto nivel para aprender y actualizar continuamente estas valoraciones dinámicas.

Para el análisis de la estructura de peones, se desarrollará un módulo especializado que identificará formaciones de peones (cadenas, islas, peones aislados, etc.) y evaluará su impacto en la posición. Este módulo considerará factores como la movilidad de los peones, su capacidad para crear peones pasados, y cómo afectan la movilidad de otras piezas. Se implementará un algoritmo de "flujo de influencia" que modelará cómo la estructura de peones afecta el control del espacio en el tablero. La evaluación de la seguridad del rey utilizará un enfoque de "mapa de calor" tridimensional. Este mapa considerará la posición del rey, la presencia de piezas defensoras cercanas, la integridad de la estructura de peones alrededor del rey, y la proximidad y coordinación de las piezas atacantes enemigas. Se implementará un algoritmo de propagación de amenazas que simulará potenciales ataques y evaluará la resistencia de la posición del rey. El análisis del control del centro, desarrollo de piezas y coordinación se realizará mediante un sistema de puntuación multifactorial. Cada pieza recibirá una puntuación basada en su desarrollo (distancia desde su posición inicial), su influencia sobre casillas centrales y estratégicas, y su coordinación con otras piezas. Se utilizará un algoritmo de optimización para encontrar la configuración ideal de piezas para una posición dada, y se comparará la posición actual con este ideal para evaluar la efectividad del desarrollo y la coordinación.

El módulo de interpretación contextual actúa como un puente entre el análisis cuantitativo del motor de ajedrez y el lenguaje natural que el LLM puede procesar eficazmente. Su función principal es transformar las evaluaciones numéricas y las variantes propuestas por el motor en descripciones cualitativas ricas en contexto. Este módulo emplea un sistema de reglas expertas combinado con técnicas de procesamiento de lenguaje natural. Comienza por

interpretar las evaluaciones numéricas, convirtiéndolas en descripciones cualitativas que reflejan la ventaja relativa de cada bando y la complejidad de la posición. Simultáneamente, analiza las variantes principales, identificando puntos críticos y generando explicaciones en lenguaje natural sobre el propósito estratégico o táctico de los movimientos clave. Para enriquecer el análisis, el módulo contextualiza la posición actual y las variantes sugeridas, comparándolas con una base de datos de posiciones clásicas y motivos tácticos/estratégicos conocidos. Esto permite al LLM relacionar el análisis actual con un contexto más amplio del conocimiento ajedrecístico. Finalmente, el módulo genera narrativas coherentes que explican el flujo de la partida y las ideas subyacentes en las variantes analizadas. Estas narrativas se estructuran utilizando un vocabulario rico en términos ajedrecísticos y conceptos estratégicos, facilitando al LLM la generación de análisis profundos y explicaciones accesibles.

CLASIFICACIÓN DE INTENCIONES, EXTRACCIÓN DE MOVIMIENTOS Y SISTEMA DE PROMPTS

El Bot Tutor de Ajedrez ofrece una experiencia de usuario intuitiva y fluida, con un enfoque principal en la interacción en lenguaje natural. El sistema emplea un enfoque híbrido para la clasificación de intenciones y la extracción de movimientos, combinando modelos de lenguaje de vanguardia con técnicas basadas en reglas. En el núcleo de este enfoque se encuentran dos componentes clave: un modelo BERT para la clasificación de intenciones y un modelo RoBERTa para la extracción de movimientos.

El modelo BERT fue ajustado con un conjunto de datos amplio y está diseñado para ser un clasificador zero-shot, lo que le permite identificar con precisión una amplia gama de intenciones del usuario, incluso en situaciones fuera de su espacio de entrenamiento. Esto habilita al sistema para comprender y responder a varios tipos de entrada del usuario, desde

comandos simples de movimiento hasta consultas estratégicas complejas. La clase *IntentClassifier* maneja la clasificación de intenciones utilizando una combinación de expresiones regulares para la detección rápida de movimientos y el modelo BERT para determinar qué tipo de interacción el usuario está tratando de expresar. Para la extracción de movimientos, el modelo RoBERTa ha sido utilizado para identificar y analizar movimientos de ajedrez expresados en lenguaje natural. A través de técnicas como aumento de datos y usando el contexto para indicar un listado de posibles movimientos legales, el modelo ha sido entrenado para manejar una variedad de formatos y notaciones de movimiento, asegurando un alto nivel de precisión y flexibilidad en el procesamiento de la entrada del usuario. El método *extract_move* utiliza el modelo RoBERTa para extraer movimientos del texto cuando las expresiones regulares no son suficientes.

Para facilitar una comunicación fluida entre el usuario y el Bot Tutor de Ajedrez, se ha implementado un flujo de interacción. La vista *send_message* actúa como el punto de entrada principal, procesando la entrada del usuario a través del clasificador de intenciones y pasando el resultado al sistema de lógica de ajedrez (*ChessLogicUnit*). Dependiendo de la intención detectada y el estado actual del juego, el sistema genera una respuesta apropiada. Central a la experiencia del usuario es el sistema de prompts y la capacidad de generación de respuestas del sistema. El Bot Tutor de Ajedrez emplea una arquitectura de dos niveles para generar respuestas, con caminos separados para manejar interacciones básicas y proporcionar análisis detallados. La clase *PromptMaker* es responsable de crear prompts apropiados basados en la intención del usuario, el historial de movimientos y el contexto de la conversación.

Para interacciones básicas, como responder a comandos simples de movimiento o consultas generales relacionadas con el ajedrez, el sistema se basa en un conjunto de plantillas

predefinidas (*create_move_prompt* y *create_chat_prompt* en *PromptMaker*). Estas plantillas están diseñadas para proporcionar respuestas concisas e informativas, asegurando una interacción rápida y eficiente. Cuando los usuarios solicitan análisis o explicaciones más profundas, el Bot Tutor de Ajedrez activa su módulo de análisis detallado. Este módulo aprovecha plantillas de prompts dinámicas que incorporan un rico conjunto de información contextual, incluyendo el estado actual del tablero, el historial del juego y conceptos relevantes de ajedrez. Al gestionar cuidadosamente este contexto y generar prompts personalizados, el sistema puede proporcionar respuestas altamente detalladas y perspicaces que se adaptan a las necesidades específicas de cada usuario. La generación de respuestas se maneja a través del método *quick_response*, que utiliza la API de Claude de Anthropic para conectarse al modelo Haiku 3.5, la última versión de un modelo neuronal relativamente pequeño y rápido lanzado durante el desarrollo del proyecto (Anthropic, 2024). Al combinar modelos de lenguaje avanzados, un flujo de interacción y un sistema de prompts, el Bot Tutor de Ajedrez ofrece una experiencia de usuario inmersiva. Los usuarios pueden interactuar con el sistema utilizando lenguaje natural, recibiendo retroalimentación personalizada, análisis y orientación para ayudarles a mejorar sus habilidades de ajedrez y profundizar su comprensión del juego.

INTEGRACION Y DESAROLLO

El desarrollo del Bot Tutor de Ajedrez fue un proceso iterativo y adaptativo, de acuerdo con la planificación de seguir una metodología ágil, con una arquitectura de sistema que evolucionó continuamente para hacer frente a los desafíos y aprovechar las oportunidades encontradas durante el proceso. La idea inicial, que consistía en una interfaz de usuario conectada a un backend de Django y un módulo de lógica de ajedrez, rápidamente se transformó en un complejo ensamblaje de componentes interconectados. Sin embargo, a medida que el proyecto avanzaba, se hizo evidente la necesidad de incorporar módulos

adicionales para abordar aspectos críticos, como el manejo de las intenciones del usuario y la generación de prompts. Esto llevó al equipo a explorar y seleccionar modelos adecuados de la plataforma Hugging Face, como BERT y RoBERTa, para cumplir con estos requisitos.

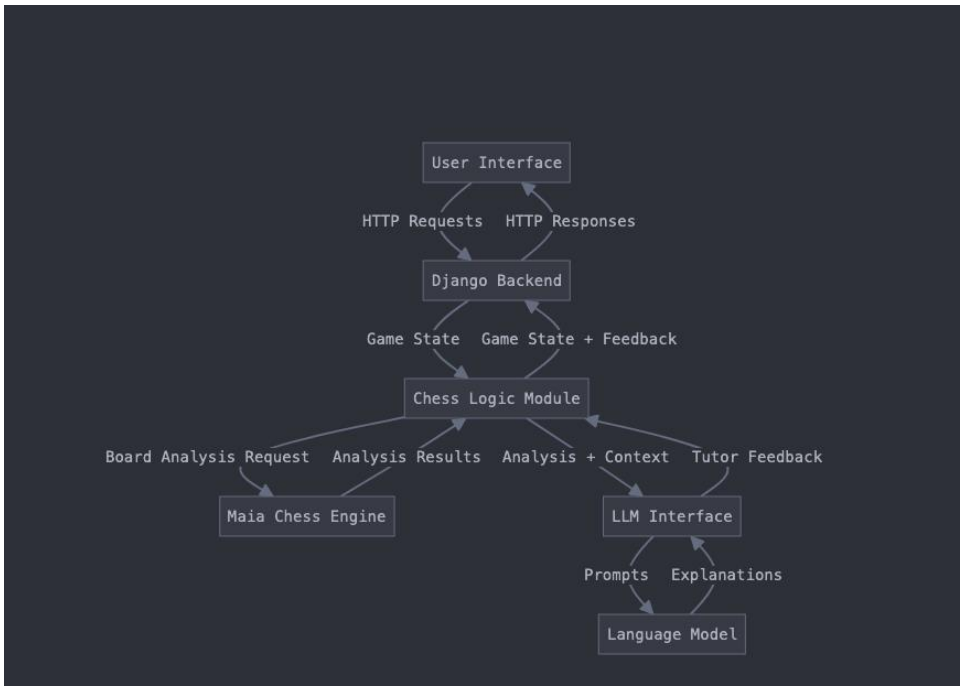

Figura 4: Vista general de los módulos del sistema y sus respectivas interacciones.

Un punto de inflexión significativo en el proceso de desarrollo fue la decisión de migrar del modelo LLaMa 3.2, un modelo de 1B parámetros ejecutado localmente en una Mac mini M2, al modelo Haiku 3.5, más eficiente y accesible a través de la API de Anthropic. Esta transición se fundamentó en los prolongados tiempos de respuesta de hasta 17 minutos y la falta de coherencia en las respuestas generadas por LLaMa. La adopción de Haiku 3.5 no solo mejoró considerablemente el rendimiento del sistema, sino que también simplificó la optimización del código para diferentes CPUs/GPUs. Aunque esta decisión implicó un costo

adicional asociado al acceso a la API, en la práctica, este costo ha sido mínimo debido a la eficiencia y bajo costo del modelo.

Durante la fase de pruebas, aunque limitada, se obtuvieron resultados significativos en varios aspectos clave del sistema. Las evaluaciones de rendimiento mostraron una mejora sustancial en la capacidad de análisis posicional, con el sistema demostrando particular aptitud en la identificación de motivos tácticos básicos y patrones posicionales comunes. La integración del motor Maia con el modelo de lenguaje demostró ser especialmente efectiva en posiciones de medio juego, donde el sistema pudo generar explicaciones contextualmente relevantes que resonaron con las evaluaciones de los jugadores más experimentados del grupo de prueba. El tiempo de respuesta del sistema mejoró dramáticamente tras la transición a Haiku 3.5, pasando de los anteriormente mencionados 17 minutos a respuestas casi instantáneas.

Las pruebas de usabilidad, aunque realizadas con un grupo reducido, revelaron patrones interesantes en la interacción usuario-sistema. Los participantes de nivel intermedio mostraron los niveles más altos de satisfacción, indicando que las explicaciones del sistema eran particularmente útiles para jugadores que ya comprenden los fundamentos del juego pero buscan profundizar su entendimiento estratégico. La capacidad del sistema para adaptar su nivel de explicación resultó especialmente efectiva para este grupo demográfico. Sin embargo, se observaron limitaciones en el manejo de consultas muy específicas sobre aperturas poco comunes, donde el sistema ocasionalmente proporcionaba explicaciones demasiado generales.

El análisis de las interacciones también reveló que el sistema era particularmente efectivo en identificar y explicar errores tácticos inmediatos, proporcionando explicaciones claras y comprensibles de por qué ciertos movimientos eran subóptimos. La integración de la

retroalimentación del motor con las explicaciones en lenguaje natural demostró ser especialmente valiosa en estas situaciones, permitiendo a los usuarios comprender no solo qué movimientos eran incorrectos, sino también por qué lo eran y cómo mejorar su proceso de toma de decisiones. Esta capacidad de proporcionar retroalimentación inmediata y contextualizada representa un avance significativo en la aplicación de sistemas de tutoría inteligente al ajedrez.

Durante el desarrollo, se enfrentó diversos desafíos, principalmente relacionados con la arquitectura del sistema y las complejidades del frontend, un área en la que no se tenía un dominio técnico avanzado. Un error particularmente desafiante, resultante de una combinación de problemas de control de versiones y una actualización de biblioteca inoportuna, llegó a consumir una semana completa de tiempo de desarrollo. A pesar de estos obstáculos, se crearon nuevos módulos según las necesidades y se integraron con los existentes de la mejor manera posible. GitHub desempeñó un papel fundamental como herramienta de colaboración y control de versiones (Martínez, 2024), manteniendo la cohesión del proyecto a medida que crecía y evolucionaba. Sin embargo, el equipo reconoce que la calidad y mantenibilidad del código podrían haber recibido mayor atención. La presión por entregar un producto funcional en ocasiones relegó la limpieza del código a un segundo plano, generando cierta deuda técnica para el futuro.

El proceso de pruebas y evaluación enfrentó desafíos debido a complicaciones en el desarrollo. Un error significativo consumió una semana de trabajo, lo que inicialmente obstaculizó los planes de realizar pruebas de usuario extensivas. No obstante, el sistema logró someterse a pruebas significativas en varios contextos. El evento Camino del Dragón proporcionó una oportunidad invaluable para probar el sistema con un grupo diverso de jugadores de ajedrez, desde principiantes hasta jugadores experimentados. Durante este evento,

el sistema demostró su capacidad para adaptarse a diferentes niveles de habilidad y estilos de juego, aunque también reveló áreas específicas que requerían mejoras, particularmente en el manejo de consultas complejas sobre estrategia avanzada.

Además de las pruebas en eventos públicos, un grupo sustancial de amigos y conocidos del equipo de desarrollo participó en sesiones de prueba prolongadas, proporcionando retroalimentación detallada sobre la interfaz de usuario y la calidad de las explicaciones generadas por el sistema. Estas sesiones resultaron particularmente valiosas para identificar patrones en la interacción usuario-sistema y refinar los algoritmos de generación de respuestas. Aunque la presentación planificada en la BYU exhibitor no pudo realizarse, las pruebas exhaustivas realizadas en preparación para este evento contribuyeron significativamente a la mejora del sistema. A pesar de las dificultades iniciales y los contratiempos en el desarrollo, se logró desarrollar un Bot Tutor de Ajedrez funcional y efectivo. La experiencia proporcionó conocimientos valiosos sobre la construcción de sistemas complejos de IA y la gestión de proyectos de desarrollo dinámicos, particularmente en el contexto de sistemas educativos interactivos. La diversidad de usuarios que probaron el sistema durante estos eventos enriqueció significativamente nuestra comprensión de cómo diferentes perfiles de jugadores interactúan con y se benefician de un tutor de ajedrez basado en IA.

CONCLUSIONES

El desarrollo del Bot Tutor de Ajedrez ha demostrado tanto el potencial como las limitaciones actuales de la integración de modelos de lenguaje grandes en sistemas educativos. En el contexto ecuatoriano, donde el acceso a instruction de alto nivel es limitado, la necesidad de herramientas educativas innovadoras es particularmente aguda. Sin embargo, el proceso de

desarrollo reveló que la creación de un sistema verdaderamente efectivo requiere más que la simple combinación de tecnologías existentes.

La transición de LLaMa 3.2 a Haiku 3.5 ejemplifica los compromisos inherentes en el desarrollo de sistemas de IA educativos. Aunque el cambio introdujo una dependencia de API externa, el costo ha sido sorprendentemente manejable, con un gasto total de aproximadamente un dólar en tokens desde su implementación. Más significativo fue el impacto en el rendimiento: la reducción en tiempos de respuesta de 17 minutos a segundos transformó fundamentalmente la experiencia del usuario. Las pruebas iniciales con usuarios revelaron limitaciones importantes en la calidad de las interacciones. Las respuestas tendían a ser formulaicas y frecuentemente carecían de contexto relevante, sugiriendo que el desarrollo de sistemas de tutoría efectivos requiere una comprensión más profunda de la pedagogía del ajedrez que la que actualmente pueden proporcionar los modelos de lenguaje.

El proceso de desarrollo también iluminó áreas críticas para futura investigación. La capacidad de seguimiento del progreso del usuario emerge como una necesidad fundamental para personalizar efectivamente la instrucción. El uso de aprendizaje por refuerzo podría permitir una identificación más precisa de las debilidades del jugador, mientras que la integración de análisis visual mejoraría la comprensión de patrones posicionales. Particularmente prometedor es el desarrollo de metodologías especializadas para diferentes aspectos del juego, adaptadas a las necesidades específicas de cada fase del aprendizaje.

Los desafíos técnicos encontrados, desde la gestión de deuda técnica hasta la integración de múltiples componentes de IA, subrayan la importancia de una planificación robusta en el desarrollo de sistemas educativos complejos. La experiencia ha demostrado que

la innovación técnica debe equilibrarse cuidadosamente con consideraciones prácticas de implementación y mantenimiento. Más allá del ajedrez, este proyecto ofrece insights valiosos para el desarrollo de sistemas de tutoría inteligente en general. La metodología y arquitectura desarrolladas proporcionan un marco de referencia para abordar el desafío fundamental de traducir conocimiento técnico complejo en instrucción comprensible y adaptativa. Este proyecto no solo representa un avance en la aplicación de IA a la enseñanza del ajedrez, sino que también establece un precedente metodológico para el desarrollo de sistemas de tutoría inteligente en otras áreas del conocimiento, abriendo nuevos caminos para la educación personalizada asistida por IA. A medida que la IA continúa evolucionando, la capacidad de crear sistemas que no solo procesen información, sino que también la comuniquen efectivamente se vuelve cada vez más crucial.

Referencias bibliográficas

Anderson, J. R., Boyle, C. F., & Reiser, B. J. (1985). Intelligent tutoring systems. Science, 228(4698), 456-462. https://doi.org/10.1126/science.228.4698.456

Anthropic. (2024, October 22). Claude 3.5 Haiku. https://www.anthropic.com/claude/haiku

Carlini, N. (2023). Chess-LLM [Computer software]. GitHub. https://github.com/carlini/chess-llm/tree/main

Kamlish, I., Jiang, R., & Barr, P. (2019). SentiMATE: Learning to play chess through natural language processing. arXiv. https://doi.org/10.48550/arXiv.1907.08321

Lai, M. (2015). Giraffe: Using deep reinforcement learning to play chess. arXiv. https://doi.org/10.48550/arXiv.1509.01549

Martinez, A. (2023). Chess-LLM-Tutor-Thesis [Computer software]. GitHub. https://github.com/TheOneWhoBurns/TAL-Chess-LLM-Tutor-Thesis

McIlroy-Young, R., Sen, S., Kleinberg, J., & Anderson, A. (2020). Aligning superhuman AI with human behavior: Chess as a model system. arXiv. https://doi.org/10.48550/arXiv.2006.01855

Saplin, M. (2024). llm_chess [Software analysis]. GitHub. https://github.com/maxim-saplin/llm_chess

Schrittwieser, J., Antonoglou, I., Hubert, T., Simonyan, K., Sifre, L., Schmitt, S., Guez, A., Lockhart, E., Hassabis, D., Graepel, T., Lillicrap, T., & Silver, D. (2020). Mastering Atari, Go, chess and shogi by planning with a learned model. arXiv. https://doi.org/10.48550/arXiv.1911.08265